CUSTOM EVENTS IN SPRING

There are number of steps to be taken to write and publish your own custom events. Follow the instructions given in this chapter to write, publish and handle Custom Spring Events.

Step	Description
1	Create a project with a name *SpringExample* and create a package *com.tutorialspoint* under the **src** folder in the created project. All the classes will be created under this package.
2	Add required Spring libraries using *Add External JARs* option as explained in the *Spring Hello World Example* chapter.
3	Create an event class, *CustomEvent* by extending **ApplicationEvent**. This class must define a default constructor which should inherit constructor from ApplicationEvent class.
4	Once your event class is defined, you can publish it from any class, let us say *EventClassPublisher* which implements *ApplicationEventPublisherAware*. You will also need to declare this class in XML configuration file as a bean so that the container can identify the bean as an event publisher because it implements the ApplicationEventPublisherAware interface.
5	A published event can be handled in a class, let us say *EventClassHandler* which implements *ApplicationListener* interface and implements *onApplicationEvent* method for the custom event.
6	Create beans configuration file *Beans.xml* under the **src** folder and a *MainApp* class which will work as Spring application.
7	The final step is to create the content of all the Java files and Bean Configuration file and run the application as explained below.

Here is the content of **CustomEvent.java** file:

```
package com.tutorialspoint;

import org.springframework.context.ApplicationEvent;

public class CustomEvent extends ApplicationEvent{

   public CustomEvent(Object source) {
      super(source);
   }

   public String toString(){
      return "My Custom Event";
   }
}
```

Following is the content of the **CustomEventPublisher.java** file:

```
package com.tutorialspoint;

import org.springframework.context.ApplicationEventPublisher;
import org.springframework.context.ApplicationEventPublisherAware;
```

```
public class CustomEventPublisher
   implements ApplicationEventPublisherAware {

   private ApplicationEventPublisher publisher;

   public void setApplicationEventPublisher
               (ApplicationEventPublisher publisher){
      this.publisher = publisher;
   }

   public void publish() {
      CustomEvent ce = new CustomEvent(this);
      publisher.publishEvent(ce);
   }
}
```

Following is the content of the **CustomEventHandler.java** file.

```
package com.tutorialspoint;

import org.springframework.context.ApplicationListener;

public class CustomEventHandler
   implements ApplicationListener<CustomEvent>{

   public void onApplicationEvent(CustomEvent event) {
      System.out.println(event.toString());
   }

}
```

Following is the content of the **MainApp.java** file:

```
package com.tutorialspoint;

import org.springframework.context.ConfigurableApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class MainApp {
   public static void main(String[] args) {
      ConfigurableApplicationContext context =
      new ClassPathXmlApplicationContext("Beans.xml");

      CustomEventPublisher cvp =
      (CustomEventPublisher) context.getBean("customEventPublisher");
      cvp.publish();
      cvp.publish();
   }
}
```

Following is the configuration file **Beans.xml**:

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

   <bean
      />

   <bean
      />

</beans>
```

Once you are done with creating source and bean configuration files, let us run the application. If everything is fine with

your application, this will print the following message:

```
My Custom Event
My Custom Event
```